# SYSTEM AND METHOD FOR CONTROLLING PROGRAM EXECUTION

## BACKGROUND OF THE INVENTION

The present invention relates to controlling the execution of a program in an information processor for processing instructions by pipelining.

In an information processor such as microprocessor or digital signal processor (DSP) for executing instructions by pipeline processing, a pipeline hazard resulting from conditional branching, i.e., a branch hazard, will happen. Particularly when there is a great number of pipeline stages, the branch hazard causes a serious problem.

To minimize the issuance of conditional branch instructions, conditionally executable instructions are adoptable effectively. For example, according to the technique disclosed in Japanese Laid-Open Publication No. 10-49368, each conditionally executable instruction contains a condition field specifying its own execution condition. That is to say, the instruction is selectively executed depending on whether or not the execution condition specified by its own condition field is satisfied. This technique is hard to apply to an information processor of the type using a short instruction word length, because each instruction to be executed by the processor cannot afford including such an additional condition field.

As for the short-word-length information processor, an instruction issued exclusively to control the conditional execution of succeeding instructions, i.e., an execution control instruction, is applicable effectively. For instance, according to the technique disclosed in Japanese Laid-Open Publication No. 7-253882, the execution control instruction contains a condition field specifying multiple registers. The number of registers specified in the condition field is always equal to the number of instructions, which succeed the execution control instruction and will be executed under controlled conditions, and is a fixed number. And it is determined based on the value of each of these registers whether or not a succeeding instruction corresponding to the register should be executed. As a result, conditional branch instructions do not have to be used so frequently.

According to the technique disclosed in the latter publication, however, if the number of instructions to be executed under controlled conditions should be increased, then the number of registers specified in the condition field of the execution control instruction has to be increased. As a result, the length of each instruction word must be increased unintentionally. Thus, the short-word-length information processor cannot control the conditional execution of so many succeeding instructions using the execution control instruction.

## SUMMARY OF THE INVENTION

An object of the present invention is controlling the conditional execution of as many succeeding instructions as possible using an execution control instruction of a short word length to suppress the branch hazard in an information processor for processing the instructions by pipelining.

To achieve this object, the present invention introduces an instruction-specifying field, which is used to define, in binary code, the number of instructions that should be executed under controlled conditions, into the execution control instruction.

More specifically, the present invention uses an execution control instruction, which contains: a condition field specifying an execution condition; and an instruction-specifying field defining, in binary code, the number of instructions to be executed only conditionally. A decision is made as to whether or not the execution condition that has been specified by the condition field is satisfied. Based on the outcome of this decision, it is determined whether or not that number of instructions, which has been defined by the instruction-specifying field for instructions succeeding the execution control instruction, should be nullified.

## BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a block diagram schematically illustrating

an exemplary configuration for a program execution control system according to the present invention.

Figure **2** illustrates a basic format for an execution control instruction according to the present invention.

Figure **3** illustrates a format for an execution control instruction according to a first embodiment of the present invention.

Figure **4** is a flowchart illustrating a program execution control procedure according to the first embodiment.

Figures **5A** and **5B** illustrate specific examples of the program execution control according to the first embodiment.

Figure **6** illustrates a format for an execution control instruction according to a second embodiment of the present invention.

Figure **7** is a flowchart illustrating a program execution control procedure according to the second embodiment.

Figures **8A** and **8B** illustrate specific examples of the program execution control according to the second embodiment.

Figure **9** illustrates a format for an execution control instruction according to a third embodiment of the present invention.

Figure **10** is a flowchart illustrating a program execution control procedure according to the third embodiment.

Figures **11A** and **11B** illustrate specific examples of the

program execution control according to the third embodiment.

Figure **12** illustrates a format for an execution control instruction according to a fourth embodiment of the present invention.

Figure **13** is a flowchart illustrating a program execution control procedure according to the fourth embodiment.

Figures **14A**, **14B**, **14C** and **14D** illustrate specific examples of the program execution control according to the fourth embodiment.

Figure **15** illustrates a format for an execution control instruction according to a fifth embodiment of the present invention.

Figure **16** is a flowchart illustrating a program execution control procedure according to the fifth embodiment.

Figures **17A**, **17B**, **17C** and **17D** illustrate specific examples of the program execution control according to the fifth embodiment.

## DETAILED DESCRIPTION OF THE INVENTION

Sub A1

Figure **1** schematically illustrates an exemplary configuration for a program execution control system according to the present invention. The system shown in Figure **1** is used for controlling the execution of a program

Cont A1

in an information processor of the type processing instructions by pipelining. As shown in Figure **1**, the system includes instruction providing section **10**, instruction executing section **20** and nullification controller **30**. The instruction providing section **10** includes program counter **11**, memory **12** and instruction register **13**. The program counter **11** sequentially specifies respective addresses of instructions to be fetched and executed. The memory **12** reads out the instructions, which together constitute a program including an execution control instruction, one after another in accordance with the addresses specified. And the instruction register **13** stores thereon the instructions read out one by one. An instruction set **INST** is provided from the instruction register **13** to the instruction executing section **20** and to the nullification controller **30**. The instruction executing section **20** includes: an arithmetic logic unit (ALU) **21** for performing arithmetic or logic operations in response to the instruction set **INST** provided; and a flag register **22** for holding multiple flags indicating the results of the computations. The nullification controller **30** receives control flags **CF** from the flag register **22**. In the illustrated embodiment, the control flags **CF** include two flags **F1** and **F2**, which have been set equal to zero or one by the ALU **21**.

Sub A2

Figure **2** illustrates a basic format for an execution control instruction according to the present invention. As

Cont A2

shown in Figure **2**, the execution control instruction contains instruction field **41**, condition field **42** and instruction-specifying field **43**. The instruction field **41** specifies the type of the instruction as an execution control instruction. The condition field **42** specifies an execution condition **EC**. And the instruction-specifying field **43** defines, in binary code, the number **N** (where N is a natural number) of instructions that will be executed only conditionally, i.e., just when the execution condition **EC** is satisfied. In response to the execution control instruction, the nullification controller **30** decides based on the control flags **CF** whether or not the execution condition **EC** specified by the condition field **42** is met. And based on the outcome of this decision, the nullification controller **30** determines whether or not the number **N** of instructions, which number has been defined by the instruction-specifying field **43** for instructions succeeding the execution control instruction, should be nullified. Suppose the nullification controller **30** has determined that the number **N** of instructions should be nullified since the execution condition **EC** is not met, the controller **30** asserts a nullification signal **NUL** to be supplied to the ALU **21**. In that case, even if the number **N** of instructions following the execution control instruction have already been executed to a midway point of the pipeline, these instructions are nullified in the ALU **21**. Accordingly,

Cont A2

the same results are attained as if NOP (no operation) instructions had been executed instead of these instructions. Alternatively, if the nullification controller **30** has determined that the number **N** of succeeding instructions should not be nullified since the execution condition **EC** is met, then the nullification signal **NUL** is not asserted. Thus, the number **N** of instructions succeeding the execution control instruction are enabled and executed by the ALU **21**.

Hereinafter, preferred embodiments of the present invention will be described with reference to Figures **3** through **17**.

EMBODIMENT 1

Figure **3** illustrates a 16-bit format of an execution control instruction according to a first embodiment of the present invention. As shown in Figure **3**, the execution control instruction contains a 12-bit instruction field **41**, a 2-bit condition field **42** and a 2-bit instruction-specifying field **43**. The instruction field **41** sets an operation code **OP** specifying the instruction type of the operation to be performed, i.e., execution control instruction. The condition field **42** specifies any of four types of execution conditions. In the illustrated embodiment, the execution condition **EC** to be specified is a binary code representing "F1 = 1", "F1 = 0", "F2 = 1" or "F2 = 0". The instruction-

specifying field **43** specifies, in binary code, the number **N** (where N=1 through 4) of instructions that will be executed under controlled conditions. In the illustrated embodiment, codes "00", "01", "10" and "11" represent "N=1", "N=2", "N=3" and "N=4", respectively.

Figure **4** illustrates a program execution control procedure according to the first embodiment. First, in Step **S1**, the execution condition **EC** and the number **N** of instructions to be executed under controlled conditions are defined for the condition field **42** and instruction-specifying field **43**, respectively. Next, in Step **S2**, the execution control instruction is executed. Then, in Step **S3**, it is determined based on the control flags **CF** whether or not the execution condition **EC** is satisfied. If the execution condition **EC** is not met, then the number **N** of instructions succeeding the execution control instruction are nullified in Step **S4**. Alternatively, if the execution condition **EC** is met, then the number **N** of instructions following the execution control instruction are executed in Step **S5**.

Figures **5A** and **5B** illustrate specific examples of the program execution control according to the first embodiment. In the illustrated examples, "F1 = 1" is the execution condition **EC** and "N = 4". As shown in Figure **5A**, if the execution condition is not met (i.e., when F1 = 0), then the four instructions succeeding the execution control

instruction (i.e., Succeeding Instructions 1 through 4) are nullified. On the other hand, if the execution condition is met (i.e., when F1 = 1), then the four instructions following the execution control instruction (i.e., Succeeding Instructions 1 through 4) are executed, not nullified, as shown in Figure **5B**.

In the first embodiment, the number of bits in the condition field **42** is two. Optionally, the number of execution conditions may be increased by increasing this bit number. The number of bits in the instruction-specifying field **43** is also two in the first embodiment. Alternatively, the number of instructions to be executed under controlled conditions may be increased by increasing this bit number. These modified examples will be equally applicable to all of the following second through fifth embodiments.

EMBODIMENT 2

Figure **6** illustrates a 16-bit format for an execution control instruction according to a second embodiment of the present invention. The format shown in Figure **6** is the same as that shown in Figure **3**.

Figure **7** illustrates a program execution control procedure according to the second embodiment. First, in Step **S11**, the execution condition **EC** and the number **N** of instructions to be executed under controlled conditions are

defined for the condition field **42** and instruction-specifying field **43**, respectively. Next, in Step **S12**, the execution control instruction is executed. Then, in Step **S13**, it is determined based on the control flags **CF** whether or not the execution condition **EC** is satisfied. If the execution condition **EC** is not met, then the number **N** of instructions succeeding the execution control instruction are nullified in Step **S14a**. Thereafter, the next number **N** of instructions following the former instruction set are executed in Step **S14b**. Alternatively, if the execution condition **EC** is met, then the number **N** of instructions succeeding the execution control instruction are executed in Step **S15a**. Thereafter, the next number **N** of instructions following the former instruction set are nullified in Step **S15b**.

Figures **8A** and **8B** illustrate specific examples of the program execution control according to the second embodiment. In the illustrated examples, "F1 = 1" is the execution condition **EC** and "N = 2". As shown in Figure **8A**, if the execution condition is not met (i.e., when F1 = 0), then the first two instructions succeeding the execution control instruction (i.e., Succeeding Instructions 1 and 2) are nullified, and the next two instructions (i.e., Succeeding Instructions 3 and 4) are executed. On the other hand, if the execution condition is met (i.e., when F1 = 1), then the first two instructions following the execution control

instruction (i.e., Succeeding Instructions 1 and 2) are executed, and the next two instructions (i.e., Succeeding Instructions 3 and 4) are nullified as shown in Figure **8B**. That is to say, according to this embodiment, instructions can be executed conditionally in accordance with the if-then-else construct.

EMBODIMENT 3

Figure **9** illustrates a 16-bit format for an execution control instruction according to a third embodiment of the present invention. As shown in Figure **9**, the execution control instruction contains a 10-bit instruction field **41**, a 2-bit condition field **42** and a 4-bit instruction-specifying field **43**. The instruction field **41** defines an operation code **OP** specifying the instruction type of the operation to be performed, i.e., execution control instruction. The condition field **42** specifies any of four types of execution conditions. The instruction-specifying field **43** consists of first and second instruction-specifying sub-fields **43a** and **43b**, each defining, in binary code, the number of instructions that will be executed only conditionally. The first instruction-specifying sub-field **43a** is a 2-bit field for defining, in binary code, the first number **N1** (where N1 = 1 through 4) of instructions to be executed under controlled conditions. The second instruction-specifying sub-field **43b** is also a 2-bit

field for defining, in binary code, the second number **N2** (where N2 = 1 through 4) of instructions to be executed under controlled conditions.

Figure **10** illustrates a program execution control procedure according to the third embodiment. First, in Step **S21**, the execution condition **EC** and the first and second numbers **N1** and **N2** of instructions to be executed under controlled conditions are defined for the condition field **42** and instruction-specifying sub-fields **43a** and **43b**, respectively. Next, in Step **S22**, the execution control instruction is executed. Then, in Step **S23**, it is determined based on the control flags **CF** whether or not the execution condition **EC** is satisfied. If the execution condition **EC** is not met, then the first number **N1** of instructions succeeding the execution control instruction are nullified in Step **S24a**. Thereafter, the second number **N2** of instructions following the first instruction set are executed in Step **S24b**. Alternatively, if the execution condition **EC** is met, then the first number **N1** of instructions succeeding the execution control instruction are executed in Step **S25a**. Thereafter, the second number **N2** of instructions following the first instruction set are nullified in Step **S25b**.

Figures **11A** and **11B** illustrate specific examples of the program execution control according to the third embodiment. In the illustrated examples, "F1 = 1" is the execution

condition **EC**, "N1 = 1" and "N2 = 3". As shown in Figure **11A**, if the execution condition is not met (i.e., when F1 = 0), then the first instruction succeeding the execution control instruction (i.e., Succeeding Instruction 1) is nullified, and the next three instructions (i.e., Succeeding Instructions 2, 3 and 4) are executed. On the other hand, if the execution condition is met (i.e., when F1 = 1), then the first instruction following the execution control instruction (i.e., Succeeding Instruction 1) is executed, and the next three instructions (i.e., Succeeding Instructions 2, 3 and 4) are nullified as shown in Figure **11B**. That is to say, according to this embodiment, instructions can be executed conditionally in accordance with the if-then-else construct. In addition, the respective numbers of instructions corresponding to the THEN and ELSE statements can be defined independently.

EMBODIMENT 4

Figure **12** illustrates a 16-bit format for an execution control instruction according to a fourth embodiment of the present invention. As shown in Figure **12**, the execution control instruction contains a 10-bit instruction field **41**, a 4-bit condition field **42** and a 2-bit instruction-specifying field **43**. The instruction field **41** defines an operation code **OP** specifying the instruction type of the operation to be

performed, i.e., execution control instruction. The condition field **42** consists of first and second condition sub-fields **42a** and **42b**, each specifying a single execution condition. The first condition sub-field **42a** is a 2-bit field for specifying any of four types of execution conditions as a first execution condition **EC1**. The second condition sub-field **42b** is also a 2-bit field for specifying any of four types of execution conditions as a second execution condition **EC2**. The instruction-specifying sub-field **43** defines, in binary code, the number **N** (where N = 1 through 4) of instructions to be executed under controlled conditions.

Figure **13** illustrates a program execution control procedure according to the fourth embodiment. First, in Step **S31**, the first and second execution conditions **EC1** and **EC2** and the number **N** of instructions to be executed under controlled conditions are defined for the first and second condition sub-fields **42a** and **42b** and instruction-specifying field **43**, respectively. Next, in Step **S32**, the execution control instruction is executed. Then, in Step **S33a**, it is determined based on the control flags **CF** whether or not the first execution condition **EC1** is satisfied. Next, in Step **S33b**, it is determined based on the control flags **CF** whether or not the second execution condition **EC2** is satisfied. If neither the first nor second execution condition **EC1** nor **EC2** is met, then the number **N** of instructions succeeding the execution control

instruction are nullified in Step **S34a**. Thereafter, the next number **N** of instructions succeeding the former instruction set are nullified in Step **S34b**. Alternatively, if the first execution condition **EC1** is not met and the second execution condition **EC2** is met, then the number **N** of instructions succeeding the execution control instruction are nullified in Step **S35a**. Thereafter, the next number **N** of instructions succeeding the former instruction set are executed in Step **S35b**. As another alternative, if the first execution condition **EC1** is met and the second execution condition **EC2** is not met, then the number **N** of instructions succeeding the execution control instruction are executed in Step **S36a**. Thereafter, the next number **N** of instructions succeeding the former instruction set are nullified in Step **S36b**. As further alternative, if both the first and second execution conditions **EC1** and **EC2** are met, then the number **N** of instructions succeeding the execution control instruction are executed in Step **S37a**. Thereafter, the next number **N** of instructions succeeding the former instruction set are executed in Step **S37b**.

Figures **14A** through **14D** illustrate specific examples of the program execution control according to the fourth embodiment. In the illustrated examples, "F1=1" is the first execution condition **EC1**, "F2 = 0" is the second execution condition **EC2** and "N=2". As shown in Figure **14A**, if neither

the first nor second execution condition is met (i.e., when F1 = 0 and F2 = 1), then four instructions succeeding the execution control instruction (i.e., Succeeding Instructions 1 through 4) are nullified. Alternatively, if the first execution condition is not met and the second execution condition is met (i.e., when F1 = 0 and F2 = 0), then two instructions succeeding the execution control instruction (i.e., Succeeding Instructions 1 and 2) are nullified and next two instructions succeeding the former instruction set (i.e., Succeeding Instructions 3 and 4) are executed as shown in Figure **14B**. As another alternative, if the first execution condition is met and the second execution condition is not met (i.e., when F1=1 and F2=1), then two instructions succeeding the execution control instruction (i.e., Succeeding Instructions 1 and 2) are executed and next two instructions succeeding the former instruction set (i.e., Succeeding Instructions 3 and 4) are nullified as shown in Figure **14C**. As further alternative, if both the first and second execution conditions are met (i.e., when F1 = 1 and F2 = 0), then four instructions succeeding the execution control instruction (i.e., Succeeding Instructions 1 through 4) are executed, not nullified, as shown in Figure **14D**.

It should be noted that the number **M** of condition sub-fields in the execution control instruction may be equal to or larger than three. In such a case, a number **N** of

instructions, which number has been defined by the instruction-specifying field **43** for instructions succeeding the execution control instruction, are regarded as instructions to be executed under controlled conditions. If an execution condition **ECm** (where m=1 through M), which has been specified by associated one of the number **M** of condition sub-fields, is not met, then the step of nullifying the number **N** of instructions at a location corresponding to the specified execution condition **ECm** is performed. Furthermore, if an execution condition **ECm** (where m=1 through M), which has been specified by associated one of the number **M** of condition sub-fields, is met, then the step of executing the number **N** of instructions at a location corresponding to the specified execution condition **ECm** is performed.

EMBODIMENT 5

Figure **15** illustrates a 16-bit format for an execution control instruction according to a fifth embodiment of the present invention. As shown in Figure **15**, the execution control instruction contains an 8-bit instruction field **41**, a 4-bit condition field **42** and a 4-bit instruction-specifying field **43**. The instruction field **41** defines an operation code **OP** specifying the instruction type of the operation to be performed, i.e., execution control instruction. The condition field **42** consists of first and second condition sub-fields

**42a** and **42b**, each specifying a single execution condition. The first condition sub-field **42a** is a 2-bit field for specifying any of four types of execution conditions as a first execution condition **EC1**. The second condition sub-field **42b** is also a 2-bit field for specifying any of four types of execution conditions as a second execution condition **EC2**. The instruction-specifying field **43** consists of first and second instruction-specifying sub-fields **43a** and **43b**, each defining, in binary code, the number of conditionally executable instructions. The first instruction-specifying sub-field **43a** is a 2-bit field for defining, in binary code, the first number **N1** (where N1 = 1 through 4) of instructions to be executed under controlled conditions. The second instruction-specifying sub-field **43b** is also a 2-bit field for defining, in binary code, the second number **N2** (where N2 = 1 through 4) of instructions to be executed under controlled conditions.

Figure **16** illustrates a program execution control procedure according to the fifth embodiment. First, in Step **S41**, the first and second execution conditions **EC1** and **EC2** and first and second numbers **N1** and **N2** of conditionally executable instructions are defined for the first and second condition sub-fields **42a** and **42b** and first and second instruction-specifying sub-fields **43a** and **43b**, respectively. Next, in Step **S42**, the execution control instruction is executed. Then, in

Step **S43a**, it is determined based on the control flags **CF** whether or not the first execution condition **EC1** is satisfied. Next, in Step **S43b**, it is determined based on the control flags **CF** whether or not the second execution condition **EC2** is satisfied. If neither the first nor second execution condition **EC1** nor **EC2** is met, then the number **N1** of instructions succeeding the execution control instruction are nullified in Step **S44a**. Thereafter, the next number **N2** of instructions succeeding the former instruction set are nullified in Step **S44b**. Alternatively, if the first execution condition **EC1** is not met and the second execution condition **EC2** is met, then the number **N1** of instructions succeeding the execution control instruction are nullified in Step **S45a**. Thereafter, the next number **N2** of instructions succeeding the former instruction set are executed in Step **S45b**. As another alternative, if the first execution condition **EC1** is met and the second execution condition **EC2** is not met, then the number **N1** of instructions succeeding the execution control instruction are executed in Step **S46a**. Thereafter, the next number **N2** of instructions succeeding the former instruction set are nullified in Step **S46b**. As further alternative, if both the first and second execution conditions **EC1** and **EC2** are met, then the number **N1** of instructions succeeding the execution control instruction are executed in Step **S47a**. Thereafter, the next number **N2** of instructions succeeding the

former instruction set are executed in Step **S47b**.

Figures **17A** through **17D** illustrate specific examples of the program execution control according to the fifth embodiment. In the illustrated examples, "F1 = 1" is the first execution condition **EC1**, "F2 = 0" is the second execution condition **EC2**, "N1 = 1" and "N2 = 3". As shown in Figure **17A**, if neither the first nor second execution condition is met (i.e., when F1 = 0 and F2 = 1), then four instructions succeeding the execution control instruction (i.e., Succeeding Instructions 1 through 4) are nullified. Alternatively, if the first execution condition is not met and the second execution condition is met (i.e., when F1=0 and F2 = 0), then one instruction succeeding the execution control instruction (i.e., Succeeding Instruction 1) is nullified and the next three instructions following the former instruction (i.e., Succeeding Instructions 2, 3 and 4) are executed as shown in Figure **17B**. As another alternative, if the first execution condition is met and the second execution condition is not met (i.e., when F1=1 and F2=1), then one instruction succeeding the execution control instruction (i.e., Succeeding Instruction 1) is executed and the next three instructions succeeding the former instruction (i.e., Succeeding Instructions 2, 3 and 4) are nullified as shown in Figure **17C**. As further alternative, if both the first and second execution conditions are met (i.e., when F1=1 and F2=

0), then four instructions succeeding the execution control instruction (i.e., Succeeding Instructions 1 through 4) are executed, not nullified, as shown in Figure **17D**.

It should be noted that the number **M** of condition sub-fields of the execution control instruction may be equal to or larger than three. In such a case, the same number **M** of instruction-specifying sub-fields, which correspond to the number **M** of condition sub-fields, respectively, are included in the instruction-specifying field of the execution control instruction. A number **Nm** of instructions (where m=1 through M), which number has been defined by associated one of the number **M** of instruction-specifying sub-fields for instructions succeeding the execution control instruction, are regarded as instructions to be executed under controlled conditions. If an execution condition **ECm** (where m = 1 through M), which is specified by associated one of the number **M** of condition sub-fields, is not met, then the step of nullifying the number **Nm** of instructions at a location corresponding to the specified execution condition **ECm** is performed. Furthermore, if the execution condition **ECm** (where m = 1 through M), which has been specified by associated one of the number **M** of condition sub-fields, is met, then the step of executing the number **Nm** of instructions at a location corresponding to the specified execution condition **ECm** is performed.